(12) NACH DEM VERTRAG ÜBER DIE INTERNATIONALE ZUSAMMENARBEIT AUF DEM GEBIET DES PATENTWESENS (PCT) VERÖFFENTLICHTE INTERNATIONALE ANMELDUNG

(19) Weltorganisation für geistiges Eigentum
Internationales Büro

(43) Internationales Veröffentlichungsdatum
12. September 2003 (12.09.2003)

PCT

(10) Internationale Veröffentlichungsnummer
WO 03/075507 A1

(51) Internationale Patentklassifikation[7]: H04L 9/26

(21) Internationales Aktenzeichen: PCT/AT03/00063

(22) Internationales Anmeldedatum:
5. März 2003 (05.03.2003)

(25) Einreichungssprache: Deutsch

(26) Veröffentlichungssprache: Deutsch

(30) Angaben zur Priorität:
A 338/2002 5. März 2002 (05.03.2002) AT

(71) Anmelder und
(72) Erfinder: **CORDES, René-Michael** [AT/AT]; Raiffeisengasse 3, A-2323 Mannswörth (AT).

(74) Anwalt: **HAFFNER, Thomas, M.**; Schottengasse 3a, A-1014 Wien (AT).

(81) Bestimmungsstaaten *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, OM, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW.

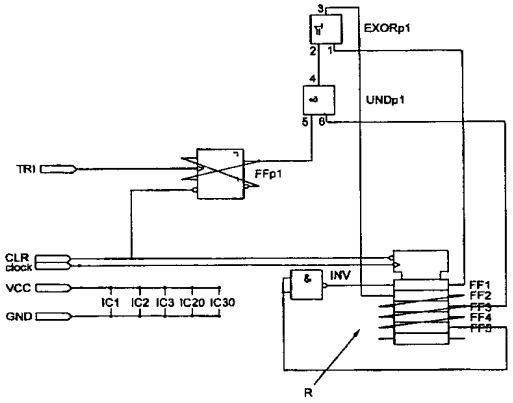
(84) Bestimmungsstaaten *(regional)*: ARIPO-Patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZM, ZW), eurasisches Patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), europäisches Patent (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IT, LU, MC, NL, PT, RO, SE, SI, SK, TR), OAPI-Patent (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Veröffentlicht:
— *mit internationalem Recherchenbericht*

[Fortsetzung auf der nächsten Seite]

(54) **Title:** CODE GENERATOR AND DEVICE FOR SYNCHRONOUS OR ASYNCHRONOUS AND PERMANENT IDENTIFICATION OR ENCODING AND DECODING OF DATA OF ANY PARTICULAR LENGTH

(54) **Bezeichnung:** CODEGENERATOR UND VORRICHTUNG ZUR SYNCHRONEN ODER ASYNCHRONEN SOWIE PERMANENTEN IDENTIKATION ODER VER- UND ENTSCHLÜSSELUNG VON DATEN BELIEBIGER LANGE



(57) **Abstract:** A code generator comprising a plurality of storage elements ($FF_{1,2,...n}$) such as flip flops which are connected to form a code-producing series (R), wherein the output of the final storage element (FF_5) in the series (R) is connected to the input of the first storage element (FF_1) in the series (R) to form a circuit and outputs and inputs of the storage elements are recursively connected by means of EXOR gates. The first input (1) of at least one EXOR gate ($EXOR_{p1}$) is connected to the output of a storage element (FF_1) disposed in the code-producing series (R), the second input (2) thereof is connected to the output of another storage element (FF_3) disposed in the code-producing series (R), and the output (3) thereof is connected to the input of the storage element (FF_2) which succeeds the storage element (FF_1) connected to the first input (1) of the EXOR gate ($EXOR_{p1}$). The output of a storage element (FF_5) disposed in the code-producing series (R) is connected to the input of an inverter (INV) and the output of the inverter (INV) is connected to the input of another storage element (FF_1) disposed in the series (R).

[Fortsetzung auf der nächsten Seite]

WO 03/075507 A1

— *vor Ablauf der für Änderungen der Ansprüche geltenden Frist; Veröffentlichung wird wiederholt, falls Änderungen eintreffen*

Zur Erklärung der Zweibuchstaben-Codes und der anderen Abkürzungen wird auf die Erklärungen ("Guidance Notes on Codes and Abbreviations") am Anfang jeder regulären Ausgabe der PCT-Gazette verwiesen.

(57) Zusammenfassung: Codegenerator mit einer Mehrzahl von zu einer codeproduzierenden Reihe (R) geschalteten Speicherelementen ($FF_{1,2,...n}$), wie z.B. Flip-Flops, wobei der Ausgang des in der Reihe (R) letzten Speicherelements (FF_5) mit dem Eingang des in der Reihe (R) ersten Speicherelements (FF_1) zu einem Kreis zusammengeschlossen ist und Ausgänge und Eingänge der Speicherelemente unter Zwischenschaltung von EXOR-Gattern rekursiv verschaltet sind. Es ist wenigstens ein EXOR-Gatter ($EXOR_{p1}$) vorgesehen, dessen erster Eingang (1) mit dem Ausgang eines in der codeproduzierenden Reihe (R) befindlichen Speicherelements (FF_1), dessen zweiter Eingang (2) mit dem Ausgang eines weiteren in der codeproduzierenden Reihe (R) befindlichen Speicherelements (FF_3) und dessen Ausgang (3) mit dem Eingang des in der codeproduzierenden Reihe (R) dem mit dem ersten Eingang (1) des EXOR-Gatters ($EXOR_{p1}$) verbundenen Speicherelement (FF_1) nachfolgenden Speicherelements (FF_2) verbunden ist. Der Ausgang eines in der codeproduzierenden Reihe (R) befindlichen Speicherelements (FF_5) ist mit dem Eingang eines Inverters (INV) und der Ausgang des Inverters (INV) mit dem Eingang eines anderen in der Reihe (R) angeordneten Speicherelements (FF_1) verbunden.

Codegenerator und Vorrichtung zur synchronen oder asynchronen sowie permanenten Identifikation oder Ver- und Entschlüsselung von Daten beliebiger Länge

Die Erfindung betrifft einen Codegenerator mit einer Mehrzahl von zu einer codeproduzierenden Reihe geschalteten Speicherelementen, wie z.B. Flip-Flops, wobei der Ausgang des in der Reihe letzten Speicherelements mit dem Eingang des in der Reihe ersten Speicherelements zu einem Kreis zusammengeschlossen ist und Ausgänge und Eingänge der Speicherelemente unter Zwischenschaltung von EXOR-Gattern rekursiv verschaltet sind.

Derartige Codegeneratoren werden zur Verschlüsselung und Übertragung von Informationen über Kommunikationsnetzwerke eingesetzt. Prinzipiell benützen alle Verschlüsselungsmethoden einen Code, auch wenn die zu verschlüsselnde Information selbst als Code Verwendung findet. Je besser der für die Verschlüsselung verwendete Code versteckt ist, desto effektiver ist die Verschlüsselung. Je länger der Code ist, desto schwerer ist er zu entschlüsseln. Beispielsweise bräuchte ein unendlicher Code gar nicht versteckt zu werden, da er ja nie ganz bekannt ist. Funktionell ist jeder Code als unendlich anzusehen, der sich nicht vor dem Ende der zu verschlüsselnden Information wiederholt. Ein funktionell unendlicher Code hat den Vorteil, dass der Ver- und Entschlüsselungsvorgang selbst denkbar einfach durch eine einzige EXOR oder EXNOR – Verknüpfung realisiert werden kann. Ein funktionell unendlicher Code hat den Nachteil, dass er nicht übertragen werden kann; er muss generiert werden.

Es gibt eine einfache Möglichkeit einen funktionell unendlichen Code zu generieren, indem man die beiden Eingänge eines EXOR-Gatters mit zwei Ausgängen von zu einer Reihe zusammengeschalteten Speicherelementen, also beispielsweise eines Schieberegisters, verbindet und den Ausgang des EXOR-Gatters mit dem Eingang des Schieberegisters rekursiv verschaltet.

Das Ergebnis ist eine Codesequenz, deren maximale Länge

$$L_c = 2^n - 1$$

(L_c = Länge der Codesequenz; n = Anzahl der codegenerierenden in Reihe geschalteten Speicherelemente)

Bit beträgt.

Nachteilig bei diesem Codegenerator ist die Tatsache, dass von der Codesequenz leicht auf die Struktur des Generators geschlossen werden kann, so dass sie mit einem gleichgebauten Generator nachgeneriert werden kann. Nachstehende Patente stellen Versuche dar, diese Codesequenzen mit weiteren Prozessen weiter zu verschlüsseln, so dass sie nicht mehr rekonstruierbar sind: US 2001033663, WO 01/05090, WO 99/22484, WO 99/16208, JP10320181, WO 98/02990 und EP 0782069. Den aus diesen Veröffentlichungen bekanntgewordenen Codegeneratoren ist gemein, dass die Länge der produzierten Codes durch Resonanzeffekte verkürzt wird. Weiters existiert eine Anzahl von Pseudozufallsgeneratoren, wie sie beispielsweise in der JP 2000-101567, JP 2001-016197, EP 1999-0913964 oder EP 1997-0782069 beschrieben sind. Diese Codegeneratoren arbeiten mit Variablen, wobei mit Hilfe eines mathematischen nichtlinearen Umrechnungsalgorithmus aus diesen Variablen eine Codesequenz errechnet wird. Dies geschieht zur Herabsetzung der Rückrechenbarkeit mit Hilfe von hohen und höchsten mathematischen Funktionen. Diesen Systemen ist gemein, dass sie sich einer mathematischen Funktionseinheit bedienen, welche einen multi-Bit-Eingang, an dem die in einem Codespeicher befindliche Ausgangsvariable anliegt, sowie einen ein-Bit-Ausgang haben, aus dem die serielle Codesequenz ausgelesen wird, was sich nachteilig auf die maximal erreichbare Codegenerierungsgeschwindigkeit auswirkt. Ebenso ist es Gegenstand der hohen und höchsten Mathematik, für komplexe Formeln einfache Lösungen zu finden, weshalb das Risiko der Entdeckung einer einfachen Lösung für eine auch noch so komplexe mathematische Funktion nie ganz auszuschließen und naturgemäß nicht abwägbar ist.

Die vorliegende Erfindung zielt darauf ab, eine Vorrichtung zur Generierung von möglichst vielen unterschiedlichen, möglichst langen Codesequenzen zu schaffen, wobei mit niedrigstem Aufwand an Schaltungselementen das Auslangen gefunden werden soll. Der Codegenerator soll unter Verwendung von Bitoperationen zur simultanen Verschlüsselung von binären Datenströmen hoher Frequenz über lange Zeiträume geeignet sein, wobei sogar der Ausgangscode

geheim bleiben soll.

Zur Lösung dieser Aufgabe besteht die vorliegende Erfindung ausgehend von einem Codegenerator der eingangs genanten Art im wesentlichen darin, dass wenigstens ein EXOR-Gatter vorgesehen ist, dessen erster Eingang mit dem Ausgang eines in der codeproduzierenden Reihe befindlichen Speicherelements, dessen zweiter Eingang mit dem Ausgang eines weiteren in der codeproduzierenden Reihe befindlichen Speicherelements und dessen Ausgang mit dem Eingang des in der codeproduzierenden Reihe dem mit dem ersten Eingang des EXOR-Gatters verbundenen Speicherelement nachfolgenden Speicherelements verbunden ist und dass der Ausgang eines in der codeproduzierenden Reihe befindlichen Speicherelements mit dem Eingang eines Inverters und der Ausgang des Inverters mit dem Eingang eines anderen in der Reihe angeordneten Speicherelements verbunden ist. Dabei ist sowohl die Struktur des Codegenerators als auch der in ihm ablaufende Algorithmus bekannt. Die Struktur ist allerdings so geartet, dass sie eine derartig hohe Anzahl an unterschiedlichen Codes in einer derartig großen Länge zu generieren im Stande ist, dass die Entdeckung des gerade verwendeten Codes so wie die aktuell produzierte Stelle in der Codesequenz nur mit einer extrem geringen Wahrscheinlichkeit möglich ist. Der Code kann dann nicht nachgeneriert werden, wenn der Generator so viele verschiedene Codes erstellen kann, dass von einem Abschnitt des einzelnen Codes nicht auf dessen Fortsetzung geschlossen werden kann. Der Generator generiert die Codesequenz auf der niedrigsten möglichen Ebene von Bit – Operationen. Es werden nicht Variablenwerte als Grundlage für die Berechnung der Codesequenzen benutzt sondern lediglich Zustände einzelner Speicherelemente, wie zum Beispiel zu einer Reihe zusammengeschaltete Flip-Flops bzw. Schieberegister. Daraus ergibt sich die höchstmögliche Effizienz im Verhältnis zur Anzahl der eingesetzten Schaltelemente einerseits und zur Gesamtlänge der generierbaren Codesequenzen sowie zur Anzahl der generierbaren verschiedenen Codes andererseits. Außerdem ist damit sichergestellt, dass der Codegenerator die höchstmögliche Produktionsgeschwindigkeit leisten kann.

Erfindungsgemäß wird die von dem Codegenerator erzeugte Codesequenz dadurch verändert, dass man zwischen zwei in der codeproduzierenden

Reihe befindliche Speicherelemente ein weiteres EXOR-Gatter einsetzt, an dessen einen Eingang man den Ausgang eines ersten Speicherelements anschließt und den zweiten Eingang vom Ausgang irgendeines weiteren in der Reihe befindlichen Speicherelements speisen lässt und schließlich mit dem Ausgang des EXOR-Gatters den Eingang des in Flussrichtung der Reihe an das erste Speicherelement anschließenden Speicherelementes speist.

Damit ausgehend von einer leeren Speicherelementen-Reihe ein Code generiert wird, der eine maximale Länge im Verhältnis zur Anzahl der verwendeten Speicherelemente aufweist, muss in der gesamten geschlossenen Reihe von Speicherelementen ein einzelner Inverter vorhanden sein. Naturgemäß kann die Funktion des Inverters mit der Funktion des EXOR-Gatters in einem Schaltelement zusammengefasst sein, beispielsweise mit Hilfe eines EXNOR-Gatters.

Um nun unterschiedliche Codes zu programmieren, wird bevorzugt das bzw. die EXOR-Gatter in ihrer rekursiven Funktion in Abhängigkeit von einem internen Codespeicherinhalt an- und abschaltbar gemacht. Zu diesem Zweck ist die Erfindung derart weitergebildet, dass in die den zweiten Eingang des wenigstens einen EXOR-Gatters und den Ausgang des weiteren in der codeproduzierenden Reihe befindlichen Speicherelements verbindende Leitung ein UND-Gatter derart geschalten ist, dass der Ausgang des UND-Gatters mit dem zweiten Eingang des EXOR-Gatters, der erste Eingang des UND-Gatters mit dem Ausgang des weiteren in der codeproduzierenden Reihe befindlichen Speicherelements und der zweite Eingang des UND-Gatters mit dem Ausgang eines der Programmierung dienenden Speicherelements verbunden ist. Der Zustand des jeweiligen der Programmierung dienenden Speicherelements bestimmt somit, ob das jeweilige EXOR-Gatter an- oder abgeschaltet ist. In der Folge wird ein derartiges Speicherelement als codeprogrammierendes Speicherelement bezeichnet.

Um den Code variantenreicher zu gestalten ist bevorzugt eine Mehrzahl von EXOR-Gattern vorgesehen, deren erster Eingang jeweils von einem Ausgang eines in der codeproduzierenden Reihe befindlichen Speicherelements gespeist wird und deren zweiter Eingang jeweils vom Ausgang eines weiteren in der codeprodu-

zierenden Reihe befindlichen Speicherelements gespeist wird, welches eine Anzahl von Speicherelementen in Flussrichtung der Reihe von dem jeweils mit dem ersten Eingang verbundenen Speicherelement entfernt ist, welche jeweils einer unterschiedlichen Primzahl entspricht, die größer als 1 und kein Teilbetrag der Gesamtzahl der in Reihe geschalteten Speicherelemente ist. Dadurch wird die Länge der produzierten Codes nicht durch Resonanzeffekte verkürzt. Dabei wird durch eine entsprechende Struktur der Eingliederung der verschiedenen codeverändernden EXOR-Gatter sichergestellt, dass zwischen den beiden Speicherelementen, welche sich in der zu einem Kreis geschlossenen codeproduzierenden Reihe und an denen sich die beiden Eingänge der EXOR-Gatter befinden, jeweils keine solchen Teilstrecken der Speicherelementen-Reihe existieren, die ein Teil oder ein Vielfaches einer anderen Teilstrecke oder der Gesamtstrecke des Kreises sind. Dies kann am effektivsten dadurch realisiert werden, dass die Anzahl der in diesen Teilstrecken befindlichen Speicherelemente sowie deren Gesamtzahl Primzahlen entsprechen.

Bei einer bevorzugten Weiterbildung der Erfindung soll der interne Codespeicherinhalt derart generiert werden, dass nicht einmal der Anwender den Inhalt des internen Codespeichers kennt. Dadurch wird die Entschlüsselung des Codes weiter erschwert. Zu diesem Zweck ist die Ausbildung bevorzugt derart getroffen, dass eine Mehrzahl von codeprogrammierenden, jeweils einem UND-Gatter und einem EXOR-Gatter zugeordneten Speicherelementen vorgesehen und in einer zu einem Kreis geschlossenen, codeprogrammierenden Reihe geschalten ist und wenigstens ein EXOR-Gatter angeordnet ist, dessen erster Eingang mit dem Ausgang eines in der codeprogrammierenden Reihe befindlichen Speicherelements, dessen zweiter Eingang mit dem Ausgang eines weiteren in der codeprogrammierenden Reihe befindlichen Speicherelements und dessen Ausgang mit dem Eingang des in der codeprogrammierenden Reihe dem mit dem ersten Eingang des EXOR-Gatters verbundenen Speicherelement nachfolgenden Speicherelements verbunden ist. Die Programmierung der Zustände der codeverändernden EXOR-Gatter mit Hilfe der UND-Gatter wird somit von einer gesonderten Speicherelementen-Reihe vorgenommen, welche in der selben Weise, wie dies bei der codeproduzierenden Speicherelementen-Reihe der Fall ist, unter Verwendung von wenigstens einem

EXOR-Gatter rekursiv verschaltet ist. Die Programmierung erfolgt dabei dadurch, dass die codeprogrammierende Speicherelementen-Reihe mit einem Programmtakt versorgt wird, wobei in einfacher Art und Weise eine Mehrzahl von Codegeneratoren auf einen identischen Code programmiert werden können, wenn, wie es einer bevorzugten Ausbildung entspricht, der Codegenerator wenigstens einen Anschluss für wenigstens einen zweiten, identisch aufgebauten Codegenerator aufweist, sodass beide Codegeneratoren zur gleichen Zeit mit dem selben Programmtakt versorgt werden können.

Die Erfindung betrifft weiters eine Einrichtung zum Senden und Empfangen von verschlüsselten Informationen mit wenigstens zwei Codegeneratoren, wobei die Codegeneratoren jeweils einen Anschluss für die gleichzeitige Versorgung der codeprogrammierenden Speicherelemente aller zusammengeschlossenen Codegeneratoren mit demselben Programmtakt aufweisen, sodass die codeprogrammierenden Speicherelemente aller zusammengeschlossenen Codegeneratoren gleichzeitig sämtliche mögliche Zustandskombinationen durchlaufen und bei gleichzeitigem Trennen der Codegeneratoren von dem Programmtakt mit derselben Programmierung versehen sind.

Bei einer bevorzugten Weiterentwicklung gemäß den Unteransprüchen 5, 6 und 9 werden auch die verschiedenen codeprogrammierenden EXOR-Gatter wiederum von einer programmierenden Speicherelementen-Reihe bestehend aus weiteren Speicherelementen in ihrer programmbeeinflussenden Wirkung ein- und ausgeschaltet, so dass bei der Programmierung nicht sämtliche möglichen Programmierungszustände linear durchlaufen werden müssen und daher auch aus der Dauer der Programmierzeit nicht einmal annähernd auf den Zustand der Schlüssel nach deren Programmierung geschlossen werden kann.

Die Erfindung wird nachfolgend anhand von in der Zeichnung dargestellten Ausführungsbeispielen näher erläutert. In dieser zeigt Fig. 1 eine Prinzipschaltung für eine programmierbare rekursive Codegeneration, Fig. 2 ein Gesamtschaltungsbeispiel für eine aus einem Codegenerator aufgebaute Funktionseinheit, mit deren zwei man eine verschlüsselte Verbindung zwischen zwei Computern aufbauen kann, und Fig. 3 ein Gesamtschaltungsbeispiel eines abgewandelten Codegenerators.

Fig. 1 zeigt eine Prinzipschaltung, bei der fünf Speicherelemente, nämlich die zu einer codeproduzierenden Reihe R zusammengeschalteten Flip-Flops $FF_{1,2,3,4,5}$, und ein EXOR-Gatter $EXOR_{p1}$ und ein UND-Gatter UND_{p1} und einen Inverter INV verschaltet sind, und zwar in der Weise, dass der Eingang 2 des EXOR-Gatters $EXOR_{p1}$ mit dem Ausgang 4 des UND-Gatters UND_{p1}, dessen einer Eingang 5 mit einem Ausgang eines der Programmierung dienenden Speicherelements FF_{p1} und dessen anderer Eingang 6 mit dem Ausgang des in der codeproduzierenden Reihe befindlichen Speicherelements FF_3, und der andere Eingang 1 des EXOR-Gatters $EXOR_{p1}$ mit dem Ausgang des in der codeproduzierenden Reihe R befindlichen Speicherelements FF_1 und der Ausgang 3 der EXOR-Gatters $EXOR_{p1}$ mit dem Eingang des Speicherelements FF_2 und der Ausgang des Speicherelements FF_5 mit den Eingängen des Inverters INV und der Ausgang des Inverters INV wiederum mit dem Eingang des in Flussrichtung nächsten Speicherelements FF_1 in der Reihe - sohin rekursiv - verbunden ist. Man erhält mit dieser Schaltung ausgehend von einer Reihe R völlig leerer Speicherelemente $FF_{1,2,3.4.5}$ eine Codesequenz generiert. Es vergehen mindestens 3 Takte ehe sich der Code wiederholt. Die einzelnen Schaltelemente können mit handelsüblichen Bausteinen verwirklicht werden: Beispielswiese kann ein IC der Type 74HC174 für die aneinandergereihten Speicherelemente $FF_{1,2,3,4,5}$ verwendet werden, ebenso ein IC 74HC08 für das UND-Gatter UND_{p1}, ein IC 74HC386 für das EXOR-Gatter $EXOR_{p1}$, ein IC 74HC00 für den Inverter INV und in IC 74HC107 für den Speicherbaustein FF_{p1}.

Die in Fig. 1 gezeigte Reihe kann naturgemäß verlängert werden und es kann sich beispielsweise eine verlängerte Reihe R ergeben, wie sie in Fig. 2 dargestellt ist. Dabei kann eine Anzahl ununterbrochen in Reihe geschalteter Speicherelemente auch in Form von Schieberegister SRG_1, SRG_2,... verwirklicht werden. Es verdoppelt sich die Länge des Codes pro hinzugefügtem Speicherelement, so dass sich die Länge des Codes wie folgt berechnet

$$L_c = 2^n - 1$$

(L_c = Länge der Codesequenz; n = Anzahl der codegenerierenden in Reihe geschalteten Speicherelemente)

Wenn diese Einheit mit einem bestimmten Takt betrieben wird gilt für die Dauer des Codes:

$$T_c = \frac{2^n - 1}{f_c}$$

(T_c = Dauer bis sich der Code wiederholt; f_c = Codegenerierungstaktfrequenz)

Mit weniger als 50 Speicherelementen bei einer Codegenerierungstaktfrequenz von 384.000 Bit/s läuft der Code länger als ein Jahr ohne dass sich die Sequenz wiederholt, so dass ein zu verschlüsselndes Signal simultan über einen ebenso langen Zeitraum verschlüsselt über eine Standleitung übersendet und entschlüsselt werden kann, so dass Übertragungen live über einen ebenso langen Zeitraum möglich sind.

Wenn man nun bei entsprechender Länge der Speicherelementen-Reihe R an mehren Stellen dieser Speicherelementen-Reihe R zwischen einem Speicherelement $FF_{1,2,3,4}$ und dem nächsten in der Reihe R befindlichen Speicherelement $FF_{2,3,4,5}$ ein EXOR-Gatter $EXOR_{p1,p2,p3,p4}$ einfügt und dieses dann mit dem Signal von einem dritten Speicherelement $FF_{8,15,20,23}$ speist, so verändert man jeweils den dadurch erzeugten Code (Fig. 2).

Bei einer Mehrzahl von codeverändernden EXOR-Gattern $EXOR_{p1,p2,p3,p4}$, siehe Fig. 2, soll sichergestellt sein, dass die verschiedenen codeverändernden EXOR-Gatter $EXOR_{p1,p2,p3,p4}$, deren erster Eingang von einem Ausgang eines Speicherelements $FF_{1,2,3,4}$ gespeist wird, ihren zweiten Eingang jeweils vom Ausgang eines Speicherelements $FF_{8,15,20,23}$ gespeist erhalten, welches eine Anzahl von Speicherelementen in Flussrichtung vom erstgenannten Speicherelement $FF_{1,2,3,4}$ entfernt ist, welche jeweils einer unterschiedlichen Primzahl entspricht, die größer als 1 aber kein Teilbetrag der Gesamtzahl der in Reihe R geschalteten Speicherelemente ist, sodass es bei der Beeinflussung der Codesequenz zu keinen codesequenzverkürzenden Resonanzeffekten kommt. Zwischen den entsprechenden

Speicherelementpaaren $FF_{1,8}$; $FF_{2,15}$; $FF_{3,20}$; $FF_{4,23}$ liegt also jeweils eine Anzahl von 7, 13, 17 und 19 (Primzahlen) Speicherelementen.

Wenn man an einen der beiden Eingänge 2 des jeweiligen EXOR-Gatters $EXOR_{p1}$ bzw. $EXOR_{p1,p2,p3,p4}$ den Ausgang 4 eines UND-Gatters UND_{p1} bzw. $UND_{p1,p2,p3,p4}$ dessen einer Eingang 6 am Ausgang des Speicherelements FF_3 bzw. $FF_{8,15,20,23}$ hängt, anschließt, dann kann man dieses EXOR-Gatter $EXOR_{p1}$ bzw. $EXOR_{p1,p2,p3,p4}$ in seiner codeverändernden Wirkung über den zweiten Eingang 5 des UND-Gatters UND_{p1} bzw. $UND_{p1,p2,p3,p4}$ an- und abschalten und wenn man daran jeweils ein weiteres Speicherelement FF_{p1} bzw. $FF_{p1,p2,p3,p4}$ anschließt, das An- und Abschalten der codebeeinflussenden Wirkung des EXOR-Gatters $EXOR_{p1}$ bzw. $EXOR_{p1,p2,p3,p4}$ programmierbar machen (Fig. 1 bzw. Fig. 2). Die codeprogrammeirenden Speicherelemente $FF_{p1,p2,p3,p4}$ können dabei zu einer Reihe RR zusammengeschaltet sein. In weiterer Folge können die codeprogrammierenden Speicherelemente $FF_{p1,p2,p3,p4}$ selbst wiederum mit Hilfe eines EXOR-Gatters $EXOR_{pp1}$ rekursiv verschaltet werden.

Die Anzahl der programmierbaren unterschiedlichen Codes berechnet sich wie folgt:

$$N_c = 2^{pn} - 1$$

(N_c = Anzahl der möglichen unterschiedlichen Code; pn = Anzahl der programmierbaren EXOR – Gatter $EXOR_{p1,p2,...pn}$)

Wenn man nun im Besitz eines identen Codegenerators ist, und an Hand einer bestimmten Anzahl von Bit den weiteren Verlauf der Codesequenz erschließen möchte so hängt die Wahrscheinlichkeit, mit der man die richtige Fortsetzung der Codesequenz erkennt, sowohl von der Anzahl der in der Codegenerierung verwendeten Speicherelemente $FF_{1,2,...n}$ als auch jener der programmierbaren, codeverändernden EXOR-Gatter $EXOR_{p1,p2,...pn}$ ab. Daraus ergibt sich eine Wahrscheinlichkeit, die dem Code zugrunde liegende Programmierung zu entdecken und sohin den weiteren Verlauf des Codes vorauszusagen von:

$$W = \frac{N_b}{(2^n - 1) * (2^{pn} - 1)}$$

(N_b = Anzahl der beobachteten Bit der Codesequenz; n = Anzahl der codegenerierenden in Reihe geschalteten Speicherelemente $FF_{1,2,...n}$; pn = Anzahl der programmierbar den Code verändernden EXOR-Gatter $EXOR_{p1,p2,...pn}$)

Beispiel:

233 ist die 52. Primzahl. Wenn man die 1 nicht nützt und die 233 die Gesamtzahl der in Reihe geschalteten Speicherelemente ausdrückt, so befinden sich auf dieser Strecke 50 unterschiedliche Speicherelemente, welche sich jeweils in Entfernung von einem Ausgangs-Speicherelement befinden, die einer Primzahl entspricht (np = 50). Da jedes rekursive EXOR-Gatter$_{1-50}$, jeweils zwischen einem nächsten Speicherelement$_{1-50}$ beginnend vom ersten in Reihe eingeschaltet ist, verlängert sich die Gesamtlänge der Speicherelemente auf (n = 233 + 50 = 283).

Daraus folgt:

$$W = \frac{N_b}{(2^n - 1) * (2^{pn} - 1)} = \frac{N_b}{(2^{283} - 1) * (2^{50} - 1)}$$

$$W = \frac{N_b}{(1{,}5541351138 * 10^{85} - 1) * (1{,}1258999068 * 10^{15} - 1)}$$

$$W \sim \frac{N_b}{1{,}7498005798 * 10^{100}}$$

Mit anderen Worten man muss die Codesequenz $1{,}7498005798 * 10^{100}$ Taktschritte lang beobachten, damit man mit der Wahrscheinlichkeit 1 eine bestimmte Sequenz entdeckt. Wenn die Taktfrequenz 384000 Hz beträgt ergibt dies eine notwendige Beobachtungszeit von $1{,}4449430312*10^{87}$ Jahren.

Indem man die codeprogrammierenden Speicherelemente ($FF_{p1,p2,p3,p4,p5,p6}$) rekursiv miteinander verschaltet, so dass sie innerhalb des Zeitintervalls

$$T\,pn = \frac{2^{pn} - 1}{f_p}$$

(T pn = Durchlaufzeit aller möglichen Programmierzustände; pn = Anzahl der Programm-Speicherelemente; f_p = Programmier-taktfrequenz)

sämtliche mögliche Zustandskombinationen durchlaufen, ergibt sich die Programmierung aus einer bestimmte Zeitspanne, in der die code-programmierenden Speicherelemente mit einem Programmtakt versorgt werden, so dass durch gleichzeitiges Ein- und Ausschalten des Programmtaktes an zwei identen Codegeneratoren (Einschaltimpuls und Ausschaltimpuls an Pin 12 von IC 10a in Schaltung Fig. 2) diese so durchgeführt werden kann, dass mehrere Codegeneratoren idente Codesequenzen generieren, der Inhalt der Programmierung aber nicht einmal den Programmierern bekannt ist.

Damit aus der Programmierdauer auch nicht annähernd die Program-mierung erschließbar ist kann die Programmierung zweistufig erfolgen. Hierzu kann eine weitere Programmierungs-Ebene hinzugefügt werden, indem das codeprogammierende EXOR-Gatter $EXOR_{pp1}$ selbst wiederum unter Zwischenschaltung eines UND-Gatters UND_{pp1} mit einer Speicherelementen-Reihe RRR verbunden und somit programmierbar gemacht wird, wobei wiederum ein EXOR-Gatter $EXOR_{ppp1}$ zur rekursiven Verschaltung der Reihe RRR verwendet wird (Fig. 3).

In der ersten Stufe der Programmierung wird der Programmierer pro-grammiert, so dass er einen Abschnitt der möglichen Programmier-zustände aufsucht, welcher dann der Ausgangspunkt für die darauf-folgende Programmierung darstellt, bei welcher sämtliche Punkte eines solchen Abschnittes aufgesucht werden können.

Ausgehend von obigen Rechenbeispiel wird dadurch gewährleistet, dass die $(2^{283}-1)*(2^{50}-1)$ verschiedenen Zustände in $2^{50}-1$ verschiedene Abschnitte zergliedert werden, von welchen einer in

der ersten Programmierphase ausgewählt wird. Dieser Auswahlvorgang erfolgt in maximal 2^{ppn}-1 Schritten (ppn = Anzahl der Primzahlen, die in der Anzahl der bei der Programmierung verwendeten Primzahlen (50) enthalten sind, also 16) Dies bedeutet, dass maximal 2^{16} Schritte erfolgen müssen, ehe sämtliche Abschnitte aufgesucht sind. Bei einer Programmiertaktfrequenz von 1 MHz ist dieser Vorgang in 0,065 Sekunden abgeschlossen. Ein Zeitraum, der wohl bei jeder Programmierung durchmessen wi rd, da er unter der Reaktionszeit des Menschen liegt, weshalb gewährleistet ist, dass aus der tatsächlich verstrichenen Programmierzeit keine Rückschlüsse auf die Programmierung der Schlüssel gezogen werden können.

Indem man nur jeden 2. Zustand eines Muttercodes einen von zwei Tochtercodes zuweist kann man aus einem zwei Codes schaffen, welche zwar verwandt aber nicht ähnlich sind.

Die Codierung des Ausgangssignals erfolgt durch ein EXNOR-Gatter (IC 17b, c, Fig. 2) in dessen einen Eingang das zu verschlüsselnde Signal und in dessen zweiten Eingang der Code eingebracht wird, so dass an dessen Ausgang das mit dem Code verschlüsselte Ausgangssignal erscheint.

Die Decodierung des Eingangssignals erfolgt durch ein EXNOR-Gatter, in dessen einen Eingang das zu entschlüsselnde Signal und in dessen zweiten Eingang der Code eingebracht wird, so dass an dessen Ausgang das mit dem Code entschlüsselte Ausgangssignal erscheint.

Durch Einsatz von wenigstens zwei solcher ident programmierter Codegeneratoren kann ein zweiter Inhaber desselben Codegenerators identifiziert werden und in weiterer Folge durch Synchronisation der Codegenerationen mit diesem eine ständige, verschlüsselte Datenverbindung aufgebaut werden, über die permanent und live Daten ausgetauscht werden können.

Da die Schaltung bei der Codegenerierung selbst keine CPU-Zeit in Anspruch nimmt, ist sie unabhängig von jedweder Hand-Shake Zeit und daher einzig und allein durch die spezifischen Schaltzeiten, der elektronischen Bauelemente, aus denen sie aufgebaut ist, in ihrer Codeproduktionsgeschwindigkeit begrenzt. Mit handelsüblichen TTL

Bauelementen sind so ohne weiteres Codeproduktionsgeschwindigkeiten im Megaherzbereich realisierbar.

Im nun folgenden Abschnitt wird der innere Ablauf der Schaltung nach Fig. 2 schrittweise dargestellt, wobei für die Funktionseinheit nach Schaltung Fig. 2 die Bezeichnung „Schlüssel" verwendet wird:

Innerer Ablauf der Schaltung		
AKTION	REAKTION	FUNKTION
1.) Der Schlüssel A wird mit seinem Ausgang für Computer und Schlüssel mit dem Eingang für Schlüssel des Schlüssels B verbunden	Sowohl in Schlüssel A als auch in Schlüssel B werden die jeweiligen Kontakteingänge auf LOW gesetzt	Die Produktion von Code wird in Schlüssel A und B eingestellt Schlüssel A wird von seinem eigenen Takt betrieben Der Takt wird von Schlüssel A auf Schlüssel B übertragen
2.) Die Riegeltaste (Riegel AUF/ Riegel ZU) des Schlüssel B wird erstmals betätigt	Das Riegel AUF – Signal wird von Schlüssel A nach Schlüssel B wietergeleitet In beiden Schlüsseln wird ein CLR – Signal abgeleitet	Sämtliche Schieberegister in den beiden Schlüsseln werden gelöscht
3.)	Der Takt wird in den Programmierteil sowohl von Schlüssel A als auch von Schlüssel B eingespeist	Die Programmierung läuft verzahnt synchron in beiden Schlüsseln, jedoch ohne direkte Übertragung des Programminhaltes von Schlüssel A nach Schlüssel B, ab.

4.) Die Riegeltaste des Schlüssel A wird abermals betätigt	Der Takt wird in den Programmierteil sowohl von Schlüssel A als auch von Schlüssel B nicht mehr eingespeist	Synchrones Stopp der Programmierung
5.) Die beiden Schlüssel werden getrennt	Sowohl in Schlüssel A als auch in Schlüssel B werden die jeweiligen Kontakteingänge auf HIGH gesetzt	Die Produktion von Code kann nun sowohl in Schlüssel A und B synchron gestartet werden.

Im nun folgenden Abschnitt wird der schrittweise Ablauf des Aufbaues einer verschlüsselten Verbindung zwischen zwei Computern mit zwei Funktionseinheiten nach Schaltung Fig. 2 dargestellt:

Übersicht der möglichen Abläufe			
MODUS	Identifikationsmodus	Synchronisationsmodus	Synchronmodus
CODE	X	X	X
ADRESSE		X	X
ZEIT			X
Identifizieren	X		
Synchronisieren	X	X	
Decodieren	X	X	X

Identifikationsmodus					
(A)	(B)	PHASE	HANDLUNGEN	SCHLÜSSEL (A)	SCHLÜSSEL (B)
-x	-x	Programmierung	Schlüssel (A) wird mit seinem Computer/ Schlüssel Ausgang mit dem Schlüsseleingang von Schlüssel (B) verbunden	Schlüssel (A) wird programmiert und wird ihm der Code C1 als Sendecode und der Code C2 als Empfangscode zugewiesen	Schlüssel (B) wird programmiert und wird ihm der Code C2 als Sendecode und der Code C1 als Empfangscode zugewiesen

0	0	Trennung	Schlüssel (A) wird aus Schlüssel (B) herausgezogen	Schlüssel (A) schweigt	Schlüssel (B) schweigt
0	0	Schlüsselaktivierung (A)		Schlüssel (A) wird mit einem Computer verbunden	
0	0			Eine E-Mail wird auf dem Computer von Schlüssel (A) geschrieben	
0 + 1000 + E — Mail	0	Codepaketabruf (A, 1000 + E-Mail)		Die E-Mail wird (live) mit Hilfe von C1 codiert, wobei eine Sequenz von 1000 Bit leeren Codes samt der Codeentstehungszeit uncodiert dem E-Mail vorangestellt wird	
0 + 1000 + E — Mail	0	Datenübertragung		Die E-Mail wird an die Homepage des Computers von Schlüssel (A) gesandt und dort ausgestellt	
0 + 1000 + E — Mail	0	Schlüsselaktivierung (B)			Schlüssel (B) wird mit einem Computer verbunden

0 + 1000 + E – Mail	0 + 1000	Codepaketabruf (B, 1000)			Eine Sequenz von 1000 Bit des Code C1 wird aus Schlüssel (B) ausgelesen und an eine Suchmaschine transferiert
0 + 1000 + E – Mail	0 + 1000				Wenn die Homepage von Schlüssel (A) entdeckt ist wird die E-Mail abgerufen
0 + 1000 + E – Mail	0 + 1000 + E– Mail	Codepaketabruf (B, E-Mail)			Schlüssel (B) decodiert (asynchron) mit Hilfe von C1 das Signal von Schlüssel (A)
0 + 1000 + E – Mail	A	Codefreifluss (B)			Schlüssel (B) generiert nunmehr laufend Code
0 + 1000 + E – Mail	A				Schlüssel (B) weiß jetzt wo Schlüssel (A) zu finden ist und kann mit ihm Verbindung aufnehmen um sich mit ihm zu synchronisieren.
Synchronisationsmodus					
B	0				Die E-Mail wird vom Server des Schlüssel (B) abgerufen

C	0 + 100 0	Codepaketabruf (B, 1000)			Schlüssel (B) vergleicht die 1000 Bit Sequenz von Schlüssel (A) mit seinem C1 und verschiebt diese so lange bis Synchronität besteht (erkennbar an den leeren 1000 Bit) decodiert (synchron) mit Hilfe von C1 das Signal von Schlüssel (A) Ergebnis: leer
D	0 + 100 0 + E — Mail	Codepaketabruf (B, E- Mail)			Schlüssel (B) decodiert (synchronisiert) mit Hilfe von C1 den Inhalt der Botschaft von (A)
E	E	Codefreifluss (A, B)			Schlüssel (B) weiß jetzt wo sich Schlüssel (A) auf der Zeitachse befindet, so dass er nunmehr synchron mit ihm Code aussenden kann und solcherart in den Synchronmodus wechseln kann.

Synchronmodus					
E + 100 0	E + 100 0			Schlüssel (A) decodiert (live) mit Hilfe von C2 das Signal von Schlüssel (B) Ergebnis: leer	Schlüssel (B) decodiert (live) mit Hilfe von C1 das Signal von Schlüssel (A) Ergebnis: leer
F + Botschaftsdauer	F + Botschaftsdauer	Datenübertragung	Ein Inhalt wird in Schlüssel (A) eingespeist	Schlüssel (A) codiert (live) mit Hilfe von C1 den Inhalt der Botschaft	Schlüssel (B) decodiert (live) mit Hilfe von C1 den Inhalt der Botschaft von (A)
G + Botschaftsdauer	G + Botschaftsdauer		Ein Inhalt wird in Schlüssel (B) eingespeist	Schlüssel (A) decodiert (live) mit Hilfe von C2 den Inhalt der Botschaft von Schlüssel (B)	Schlüssel (B) codiert (live) mit Hilfe von C2 den Inhalt der Botschaft

P A T E N T A N S P R Ü C H E:

1. Codegenerator mit einer Mehrzahl von zu einer codeproduzierenden Reihe (R) geschalteten Speicherelementen ($FF_{1,2,...n}$), wie z.B. Flip-Flops, wobei der Ausgang des in der Reihe (R) letzten Speicherelements (FF_5) mit dem Eingang des in der Reihe (R) ersten Speicherelements (FF_1) zu einem Kreis zusammengeschlossen ist und Ausgänge und Eingänge der Speicherelemente unter Zwischenschaltung von EXOR-Gattern rekursiv verschaltet sind, dadurch gekennzeichnet, dass wenigstens ein EXOR-Gatter ($EXOR_{p1}$) vorgesehen ist, dessen erster Eingang (1) mit dem Ausgang eines in der codeproduzierenden Reihe (R) befindlichen Speicherelements (FF_1), dessen zweiter Eingang (2) mit dem Ausgang eines weiteren in der codeproduzierenden Reihe (R) befindlichen Speicherelements (FF_3) und dessen Ausgang (3) mit dem Eingang des in der codeproduzierenden Reihe (R) dem mit dem ersten Eingang (1) des EXOR-Gatters ($EXOR_{p1}$) verbundenen Speicherelement (FF_1) nachfolgenden Speicherelements (FF_2) verbunden ist und dass der Ausgang eines in der codeproduzierenden Reihe (R) befindlichen Speicherelements (FF_5) mit dem Eingang eines Inverters (INV) und der Ausgang des Inverters (INV) mit dem Eingang eines anderen in der codeproduzierenden Reihe (R) angeordneten Speicherelements (FF_1) verbunden ist.

2. Codegenerator nach Anspruch 1, dadurch gekennzeichnet, dass in die den zweiten Eingang (2) des wenigstens einen EXOR-Gatters ($EXOR_{p1}$) und den Ausgang des weiteren in der codeproduzierenden Reihe (R) befindlichen Speicherelements (FF_3) verbindende Leitung ein UND-Gatter (UND_{p1}) derart geschalten ist, dass der Ausgang (4) des UND-Gatters (UND_{p1}) mit dem zweiten Eingang (2) des EXOR-Gatters ($EXOR_{p1}$), der erste Eingang (6) des UND-Gatters (UND_{p1}) mit dem Ausgang des weiteren in der codeproduzierenden Reihe (R) befindlichen Speicherelements (FF_3) und der zweite Eingang (5) des UND-Gatters (UND_{p1}) mit dem Ausgang eines codeprogrammierenden Speicherelements (FF_{p1}) verbunden ist.

3. Codegenerator nach Anspruch 1 oder 2, dadurch gekennzeichnet, dass eine Mehrzahl von EXOR-Gattern ($EXOR_{p1,p2,p3,p4}$) vorgesehen ist, deren erster Eingang jeweils von einem Ausgang eines in der

codeproduzierenden Reihe (R) befindlichen Speicherelements ($FF_{1,2,3,4}$) gespeist wird und deren zweiter Eingang jeweils vom Ausgang eines weiteren in der codeproduzierenden Reihe (R) befindlichen Speicherelements ($FF_{8,15,20,23}$) gespeist wird, welches eine Anzahl von Speicherelementen in Flussrichtung der Reihe (R) von dem jeweils mit dem ersten Eingang verbundenen Speicherelement ($FF_{1,2,3,4}$) entfernt ist, welche jeweils einer unterschiedlichen Primzahl entspricht, die größer als 1 und kein Teilbetrag der Gesamtzahl der in Reihe (R) geschalteten Speicherelemente ($FF_{1,2,...n}$) ist.

4. Codegenerator nach Anspruch 1, 2 oder 3, dadurch gekennzeichnet, dass eine Mehrzahl von codeprogrammierenden, jeweils einem UND-Gatter ($UND_{p1,p2,p3,p4}$) und einem EXOR-Gatter ($EXOR_{p1,p2,p3,p4}$) zugeordneten Speicherelementen ($FF_{p1,p2,p3,p4,...pn}$) vorgesehen und in einer zu einem Kreis geschlossenen Reihe (RR) geschalten ist und wenigstens ein EXOR-Gatter ($EXOR_{pp1}$) angeordnet ist, dessen erster Eingang mit dem Ausgang eines in der codeprogrammierenden Reihe (RR) befindlichen Speicherelements (FF_{p6}), dessen zweiter Eingang mit dem Ausgang eines weiteren in der codeprogrammierenden Reihe (RR) befindlichen Speicherelements (FF_{p5}) und dessen Ausgang mit dem Eingang des in der codeprogrammierenden Reihe (RR) dem mit dem ersten Eingang des EXOR-Gatters ($EXOR_{pp1}$) verbundenen Speicherelement (FF_{p6}) nachfolgenden Speicherelements (FF_{p1}) verbunden ist.

5. Codegenerator nach Anspruch 4, dadurch gekennzeichnet, dass in die den zweiten Eingang des wenigstens einen EXOR-Gatters ($EXOR_{pp1}$) und den Ausgang des weiteren in der codeprogrammierenden Reihe (RR) befindlichen Speicherelements (FF_{p3}) verbindende Leitung ein UND-Gatter (UND_{pp1}) derart geschalten ist, dass der Ausgang des UND-Gatters (UND_{pp1}) mit dem zweiten Eingang des EXOR-Gatters ($EXOR_{pp1}$), der erste Eingang des UND-Gatters (UND_{pp1}) mit dem Ausgang des weiteren in der codeprogrammierenden Reihe (RR) befindlichen Speicherelements (FF_{p3}) und der zweite Eingang des UND-Gatters (UND_{pp1}) mit dem Ausgang eines der Programmierung der codeprogrammierenden Reihe (RR) dienenden Speicherelements (FF_{pp5}) verbunden ist.

6. Codegenerator nach Anspruch 5, dadurch gekennzeichnet, dass

eine Mehrzahl von der Programmierung der codeprogrammierenden Reihe (RR) dienenden, jeweils einem UND-Gatter (UND_{pp1}) und einem EXOR-Gatter ($EXOR_{pp1}$) zugeordneten Speicherelementen ($FF_{pp1,pp2,pp3,pp4,...ppn}$) vorgesehen und in einer zu einem Kreis geschlossenen Reihe (RRR) geschalten ist und wenigstens ein EXOR-Gatter ($EXOR_{ppp1}$) angeordnet ist, dessen erster Eingang mit dem Ausgang eines in der Reihe (RRR) befindlichen Speicherelements (FF_{pp1}), dessen zweiter Eingang mit dem Ausgang eines weiteren in der Reihe (RRR) befindlichen Speicherelements (FF_{pp3}) und dessen Ausgang mit dem Eingang des in der Reihe (RRR) dem mit dem ersten Eingang des EXOR-Gatters ($EXOR_{ppp1}$) verbundenen Speicherelement (FF_{pp1}) nachfolgenden Speicherelements (FF_{pp2}) verbunden ist.

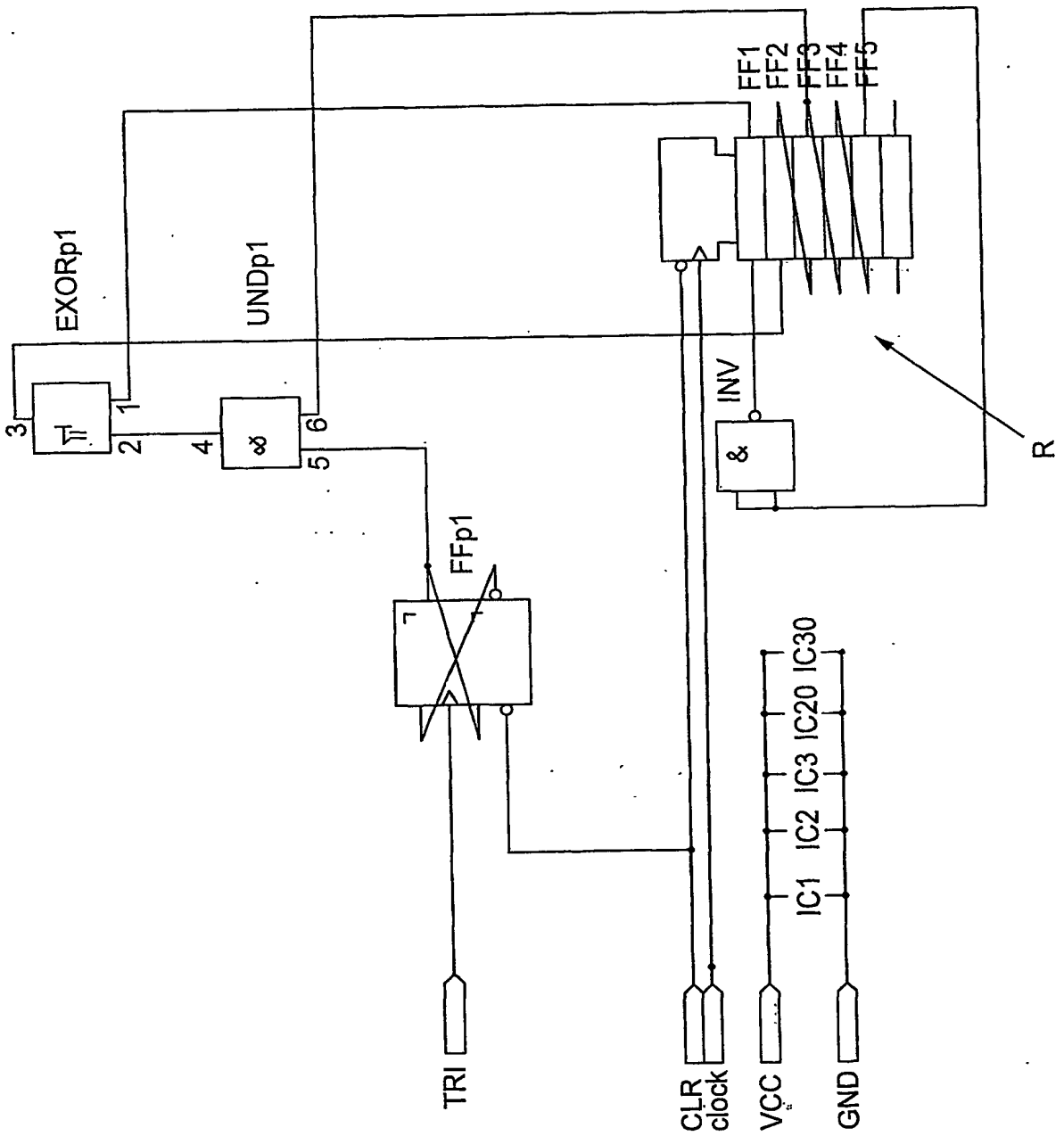
7. Codegenerator nach einem der Ansprüche 1 bis 6, dadurch gekennzeichnet, dass er wenigstens einen Anschluss für wenigstens einen zweiten, identisch aufgebauten Codegenerator aufweist, sodass beide Codegeneratoren zur gleichen Zeit mit dem selben Programmtakt versorgt werden können.

8. Einrichtung zum Senden und Empfangen von verschlüsselten Informationen mit wenigstens zwei Codegeneratoren nach einem der Ansprüche 1 bis 7, dadurch gekennzeichnet, dass die Codegeneratoren jeweils wenigstens einen Anschluss für die gleichzeitige Versorgung der codeprogrammierenden Speicherelemente ($FF_{p1,p2,p3,p4}$) aller zusammengeschlossenen Codegeneratoren mit demselben Programmtakt aufweisen, sodass die codeprogrammierenden Speicherelemente ($FF_{p1,p2,p3,...pn}$) aller zusammengeschlossenen Codegeneratoren gleichzeitig sämtliche mögliche Zustandskombinationen durchlaufen und bei gleichzeitigem Trennen der Codegeneratoren von dem Programmtakt mit derselben Programmierung versehen sind.

9. Einrichtung nach Anspruch 8, dadurch gekennzeichnet, dass die Codegeneratoren jeweils zwei Anschlüsse für die gleichzeitige Versorgung der codeprogrammierenden Speicherelemente ($FF_{p1,p2,p3,...pn}$) und der der Programmierung der codeprogrammierenden Speicherelemente ($FF_{p1,p2,p3,...pn}$) dienenden Speicherelemente ($FF_{pp1,pp2,pp3,...ppn}$) aller zusammengeschlossenen Codegeneratoren mit zwei unabhängig laufenden Programmtakten aufweisen, wobei die der Programmierung der codeprogrammierenden Speicherelemente

($FF_{p1,p2,p3,...pn}$) dienenden Speicherelemente ($FF_{pp1,pp2,pp3,...ppn}$) sämtliche mögliche Zustandskombinationen mindestens einmal und die codeprogrammierenden Speicherelemente ($FF_{p1,p2,p3,...pn}$) aller zusammengeschlossenen Codegeneratoren gleichzeitig eine bestimmte Anzahl sämtlicher möglicher Zustandskombinationen durchlaufen und nach gleichzeitigem Trennen der Codegeneratoren von den Programmtakten alle zusammengeschlossenen Codegeneratoren mit derselben Programmierung versehen sind.

Fig. 1


FF1
FF2
FF3
FF4
FF5
EXORp1
UNDp1
INV
R
FFp1
TRI
CLR
clock
VCC
GND
IC1
IC2
IC3
IC20
IC30

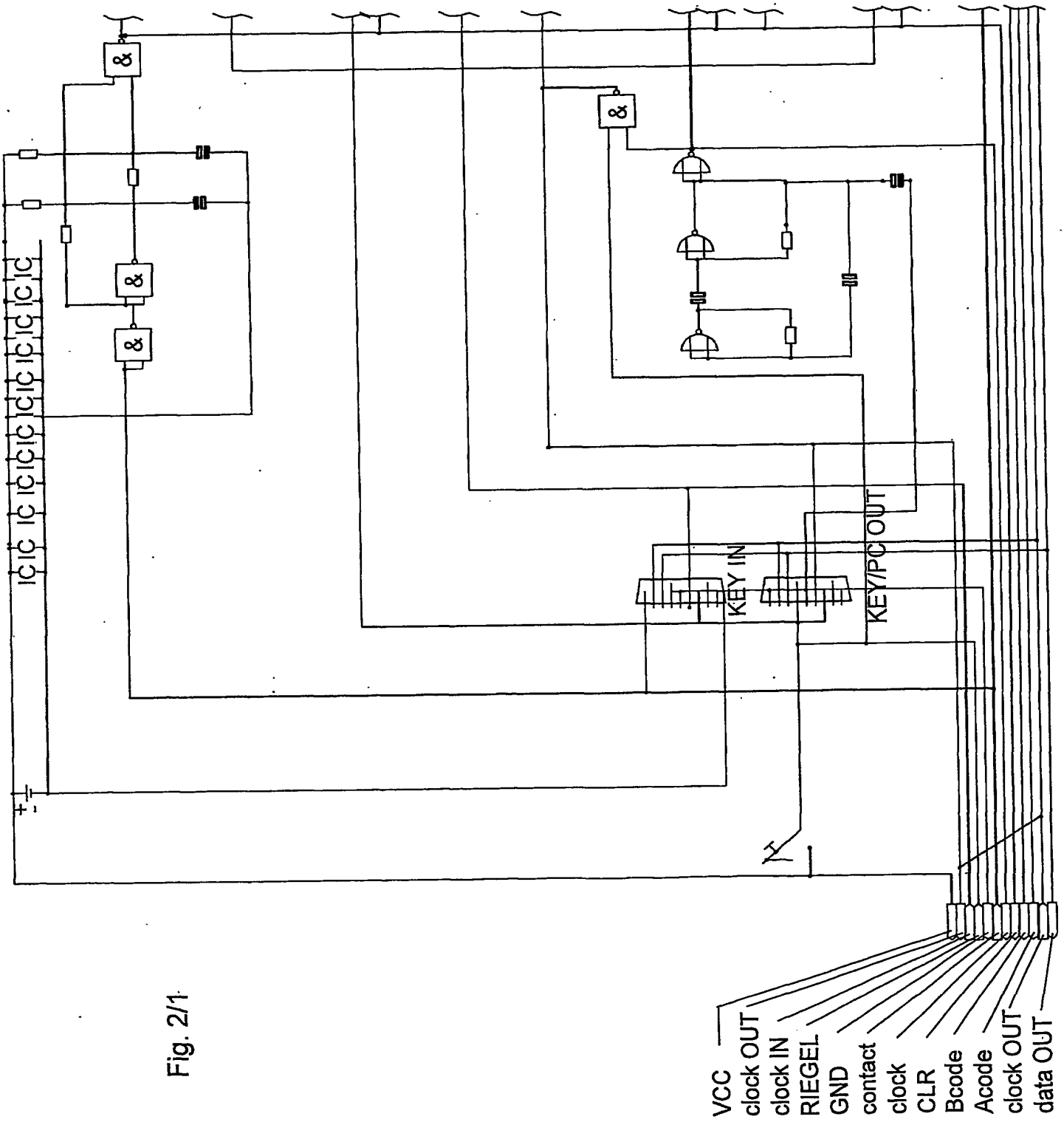
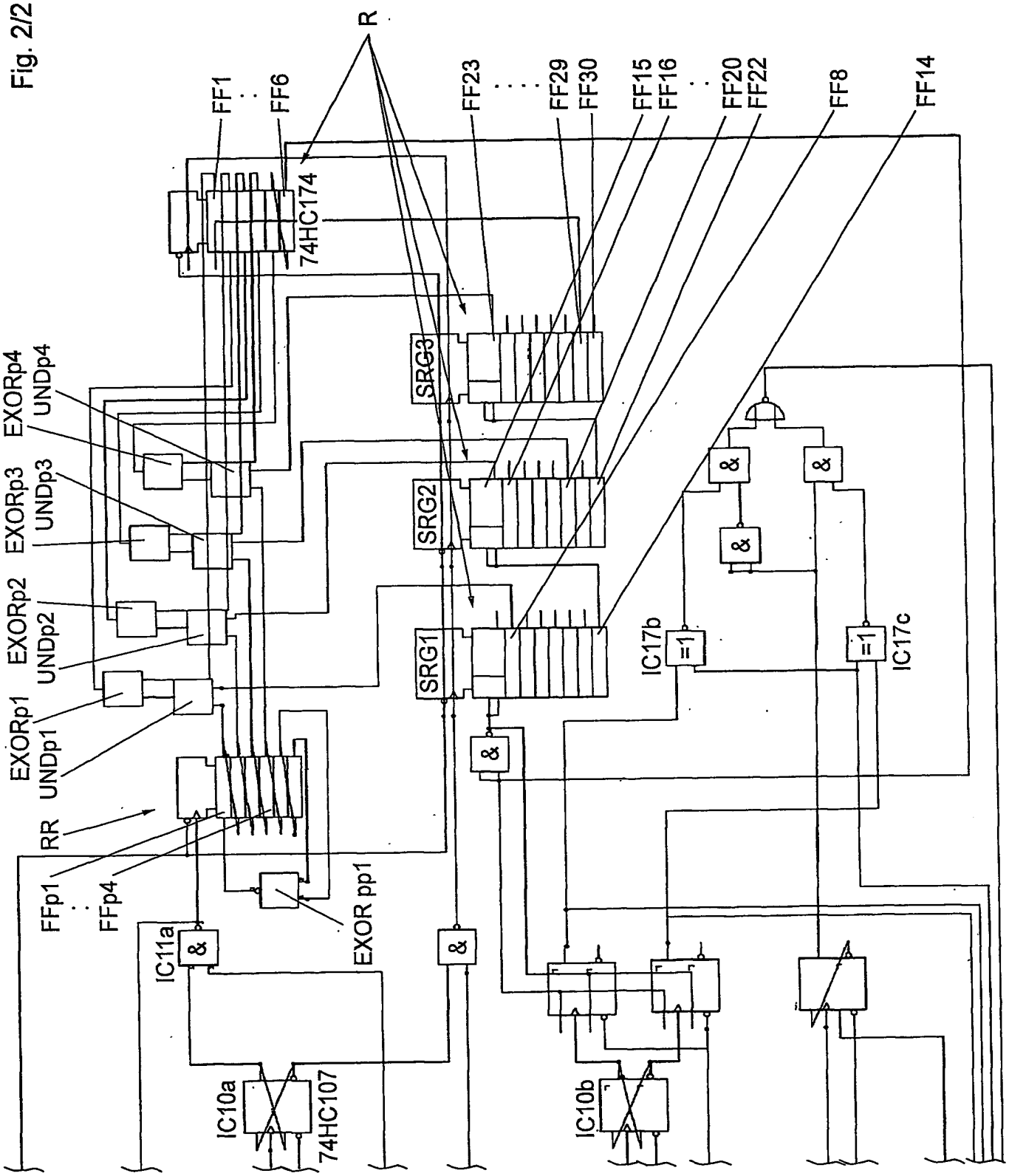

&
IC
KEY IN
KEY/PC OUT
VCC
clock OUT
clock IN
RIEGEL
GND
contact
clock
CLR
Bcode
Acode
clock OUT
data OUT

Fig. 2/1

Fig. 2/2


R
FF1
FF6
FF23
FF29
FF30
FF15
FF16
FF20
FF22
FF8
FF14
74HC174
SRG3
SRG2
SRG1
EXORp4
UNDp4
EXORp3
UNDp3
EXORp2
UNDp2
EXORp1
UNDp1
RR
FFp1
FFp4
EXOR pp1
IC11a
IC10a
74HC107
IC10b
IC17b
IC17c
=1
&

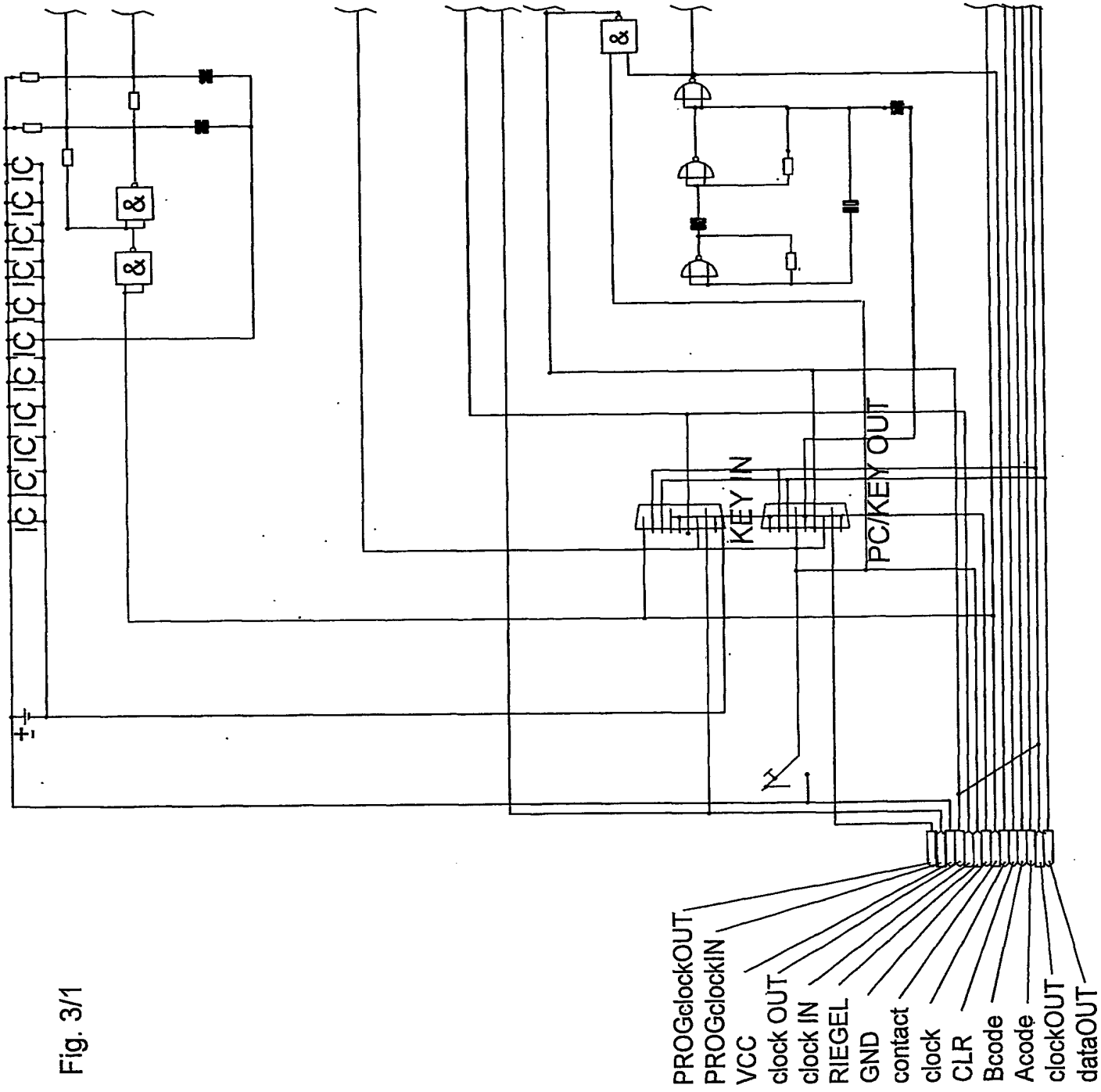
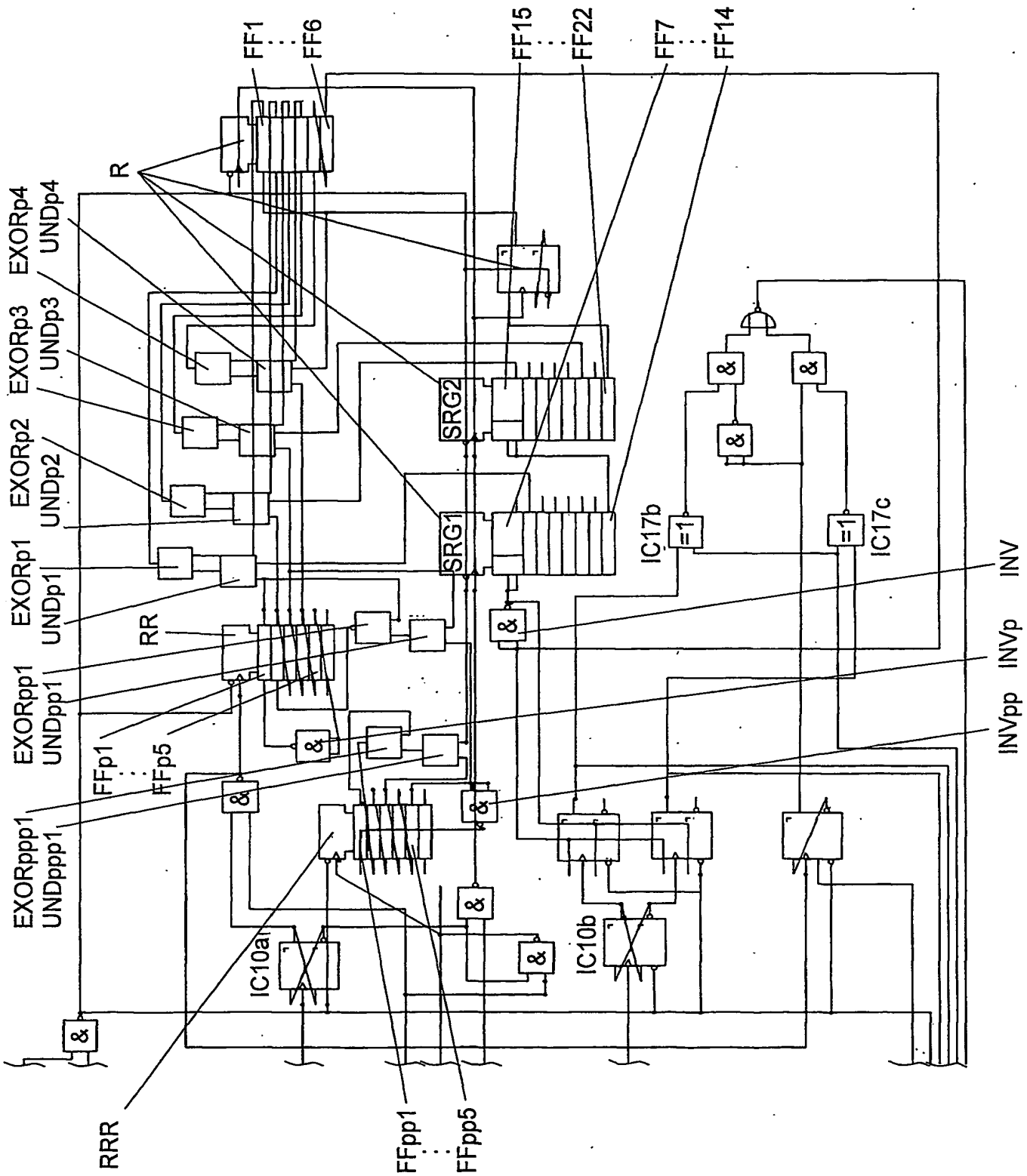

IC IC IC IC IC IC IC IC IC IC
&
&
&
KEY IN
PC/KEY OUT
PROGclockOUT
PROGclockIN
VCC
clock OUT
clock IN
RIEGEL
GND
contact
clock
CLR
Bcode
Acode
clockOUT
dataOUT

Fig. 3/1

Fig. 3/2


FF1
FF6
FF15
FF22
FF7
FF14
R
EXORp4
UNDp4
EXORp3
UNDp3
EXORp2
UNDp2
EXORp1
UNDp1
EXORpp1
UNDpp1
EXORppp1
UNDppp1
RR
RRR
FFp1
FFp5
FFpp1
FFpp5
SRG2
SRG1
IC17b
IC17c
IC10a
IC10b
=1
&
INV
INVp
INVpp

INTERNATIONAL SEARCH REPORT

International Application No
PCT/AT 03/00063

A. CLASSIFICATION OF SUBJECT MATTER
IPC 7 H04L9/26

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 7 H04L

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)
WPI Data, PAJ, EPO-Internal, IBM-TDB, INSPEC

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	KENCHENG ZENG: "PSEUDORANDOM BIT GENERATORS IN STREAM-CIPHER CRYPTOGRAPHY" COMPUTER, IEEE COMPUTER SOCIETY, LONG BEACH., CA, US, US, vol. 24, no. 2, 1 February 1991 (1991-02-01), pages 8-17, XP000219462 ISSN: 0018-9162 page 10, middle column, last line -right-hand column, line 29; figure 6	1

☐ Further documents are listed in the continuation of box C.

☐ Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
1 July 2003	11/07/2003

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL – 2280 HV Rijswijk Tel. (+31–70) 340–2040, Tx. 31 651 epo nl, Fax: (+31–70) 340–3016	Holper, G

Form PCT/ISA/210 (second sheet) (July 1992)

INTERNATIONALER RECHERCHENBERICHT

Internationales Aktenzeichen
PCT/AT 03/00063

A. KLASSIFIZIERUNG DES ANMELDUNGSGEGENSTANDES
IPK 7 H04L9/26

Nach der Internationalen Patentklassifikation (IPK) oder nach der nationalen Klassifikation und der IPK

B. RECHERCHIERTE GEBIETE

Recherchierter Mindestprüfstoff (Klassifikationssystem und Klassifikationssymbole)
IPK 7 H04L

Recherchierte aber nicht zum Mindestprüfstoff gehörende Veröffentlichungen, soweit diese unter die recherchierten Gebiete fallen

Während der internationalen Recherche konsultierte elektronische Datenbank (Name der Datenbank und evtl. verwendete Suchbegriffe)
WPI Data, PAJ, EPO-Internal, IBM-TDB, INSPEC

C. ALS WESENTLICH ANGESEHENE UNTERLAGEN

Kategorie°	Bezeichnung der Veröffentlichung, soweit erforderlich unter Angabe der in Betracht kommenden Teile	Betr. Anspruch Nr.
X	KENCHENG ZENG: "PSEUDORANDOM BIT GENERATORS IN STREAM-CIPHER CRYPTOGRAPHY" COMPUTER, IEEE COMPUTER SOCIETY, LONG BEACH., CA, US, US, Bd. 24, Nr. 2, 1. Februar 1991 (1991-02-01), Seiten 8-17, XP000219462 ISSN: 0018-9162 Seite 10, mittlere Spalte, letzte Zeile -rechte Spalte, Zeile 29; Abbildung 6	1

☐ Weitere Veröffentlichungen sind der Fortsetzung von Feld C zu entnehmen

☐ Siehe Anhang Patentfamilie

° Besondere Kategorien von angegebenen Veröffentlichungen :
"A" Veröffentlichung, die den allgemeinen Stand der Technik definiert, aber nicht als besonders bedeutsam anzusehen ist
"E" älteres Dokument, das jedoch erst am oder nach dem internationalen Anmeldedatum veröffentlicht worden ist
"L" Veröffentlichung, die geeignet ist, einen Prioritätsanspruch zweifelhaft erscheinen zu lassen, oder durch die das Veröffentlichungsdatum einer anderen im Recherchenbericht genannten Veröffentlichung belegt werden soll oder die aus einem anderen besonderen Grund angegeben ist (wie ausgeführt)
"O" Veröffentlichung, die sich auf eine mündliche Offenbarung, eine Benutzung, eine Ausstellung oder andere Maßnahmen bezieht
"P" Veröffentlichung, die vor dem internationalen Anmeldedatum, aber nach dem beanspruchten Prioritätsdatum veröffentlicht worden ist
"T" Spätere Veröffentlichung, die nach dem internationalen Anmeldedatum oder dem Prioritätsdatum veröffentlicht worden ist und mit der Anmeldung nicht kollidiert, sondern nur zum Verständnis des der Erfindung zugrundeliegenden Prinzips oder der ihr zugrundeliegenden Theorie angegeben ist
"X" Veröffentlichung von besonderer Bedeutung; die beanspruchte Erfindung kann allein aufgrund dieser Veröffentlichung nicht als neu oder auf erfinderischer Tätigkeit beruhend betrachtet werden
"Y" Veröffentlichung von besonderer Bedeutung; die beanspruchte Erfindung kann nicht als auf erfinderischer Tätigkeit beruhend betrachtet werden, wenn die Veröffentlichung mit einer oder mehreren anderen Veröffentlichungen dieser Kategorie in Verbindung gebracht wird und diese Verbindung für einen Fachmann naheliegend ist
"&" Veröffentlichung, die Mitglied derselben Patentfamilie ist

Datum des Abschlusses der internationalen Recherche	Absendedatum des internationalen Recherchenberichts
1. Juli 2003	11/07/2003

Name und Postanschrift der Internationalen Recherchenbehörde	Bevollmächtigter Bediensteter
Europäisches Patentamt, P.B. 5818 Patentlaan 2 NL – 2280 HV Rijswijk Tel. (+31–70) 340–2040, Tx. 31 651 epo nl, Fax: (+31–70) 340–3016	Holper, G